

Table based KNN for Combination of Text Summarization with Text Categorization

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based KNN version for automating the text summarization by introducing the text classification. In the previous work, we proposed the table based KNN version as an approach to the text summarization, but must present the domain as a tag, manually. In this research, we modify the KNN algorithm into the version which receives a table as the input data and apply it for the text summarization and the text classification. In the proposed system, only a text without its domain is given as the input, and the domain is assigned to the text by the text classification, and each paragraph is classified into one of summary or non-summary. The goal of this research is to reinforce the automation of the text summarization by introducing the text classification, and to improve the performance of both tasks using the table based KNN.

I. INTRODUCTION

The text summarization is the process of selecting and extracting the essential part of a text as its summary. The essential part is assumed to be some paragraphs in the text; the text summarization is interpreted into a binary classification of paragraphs into summary or non-summary. The process of text summarization is to partition a text which is given as the input into paragraphs, classify them into one of the two paragraphs, and extract paragraphs which are classified into summary. The classification task which is mapped from the text summarization is a domain dependent task where a same paragraph may be classified differently depending on the domain. It is required to present the domain as a tag for executing the text summarization.

This research is motivated by the fact that the domain is presented for summarizing a text. The text summarization is set as an independent binary classification domain by domain. An independent classifier is allocated to each domain and we need to know the domain in advance for nominating the corresponding classifier. The process of assigning a domain to a text before doing the text summarization is viewed into an instance of the text classification. This research is intended to connect the text summarization with the text classification for solving the problem.

We adopt the table based KNN algorithm for implementing the text summarization by connecting it with the text classification. We define the similarity metric between tables and modify the KNN algorithm into the version which processes tables directly. The table-based version is applied

to the text categorization whose role is to assign a domain to an input text. In the process of text summarization, a text is partitioned into paragraphs, each paragraph is classified into summary or non-summary by the table based KNN algorithm, and paragraphs which are classified into summary are extracted. In this research, we propose the table based KNN algorithm as the approach to both tasks, the text summarization and the text classification.

Let us mention some benefits from this research. The problems in encoding texts into numerical vectors is solved by encoding them into tables as another type of structured form. The performance of the text summarization and the text classification is improved by adopting the table based KNN algorithm as the approach to the two tasks. There is no need of presenting a domain manually for every input text in using the text summarization system by connecting the text summarization with the text classification. We automate the process of decoding the domain through the text classification in the text summarization system, as its upgrade.

Let us mention remaining tasks for reinforcing this research. We may define additional operations on tables. Paragraphs and texts are encoded into compound structured forms, each of which consists of more than one structured form. Other machine learning algorithms and deep learning algorithms may be modified into table-based versions. The text summarization system which relates to the text classification module may be implemented as a real program or a library by adopting the table based KNN algorithm.

II. PROPOSED WORKS

A. Encoding Process

This section is concerned with the table as the text representation. It is defined as a set of entries, each of which consists of an element and its weight. The table which represents a text consists of entries, each of which is a pair of a word and its weight in the text. The similarity between tables is computed based on shared words as the semantic similarity between texts. This section is intended to describe the process of encoding a text into a table and the process of computing the similarity between tables.

The process of encoding a text into a table is illustrated in Figure 1. It is indexed into a list of words. The weights of

words in the text are computed in the text-word weighting; the word weight in the text is a relationship between a word and a text. The entries with their lower weights are removed for downsizing the table. Refer to [1] for studying the entire process in more detail.

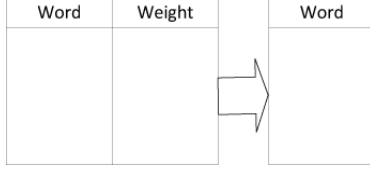


Figure 1. Process of Encoding Texts into Tables

Let us mention the function of a table which maps it into a set of words as shown in Figure 2. The table which represents a text is expressed as entry set as shown in equation (1),

$$T = \{(t_1, w_1), (t_2, w_2), \dots, (t_{|T|}, w_{|T|})\} \quad (1)$$

where t_i is a word, and w_i is the weight of the word, t_i , in the text. The function for generating a list of words is expressed as equation (2),

$$F(T) = \{t_1, t_2, \dots, t_{|T|}\} \quad (2)$$

The function is the operation which takes only words without their weights as a set. The operation is applied to computing the similarity between tables which represent texts.

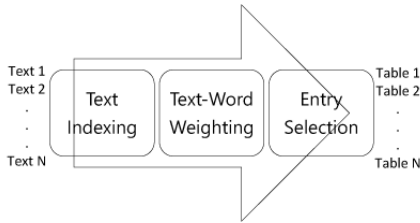


Figure 2. Conversion of Table into Word Set

Let us mention the process of computing the similarity between two tables as the similarity between two texts. The tables which represent texts are notated by the two sets: $D_1 = \{(t_{11}, w_{11}), (t_{12}, w_{12}), \dots, (t_{1|D_1|}, w_{1|D_1|})\}$ and $D_2 = \{(t_{21}, w_{21}), (t_{22}, w_{22}), \dots, (t_{2|D_2|}, w_{2|D_2|})\}$. The intersection between the two sets which are generated by applying the function, $F(\cdot)$, by equation (3),

$$F(D_1) \cap F(D_2) = \{t_{s1}, t_{s2}, \dots, t_{s|F(D_1) \cap F(D_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared word, its weight from the table, D_1 , its weight from the table, D_2 , is constructed based on equation (3), as expressed in equation (4),

$$SD_{12} = \{(t_{s1}, w_{s1}^1, w_{s1}^2), (t_{s2}, w_{s2}^1, w_{s2}^2), \dots, (t_{s|F(D_1) \cap F(D_2)|}, w_{s|F(D_1) \cap F(D_2)|}^1, w_{s|F(D_1) \cap F(D_2)|}^2)\} \quad (4)$$

The similarity between the two tables, D_1 and D_2 , is computed by equation (5),

$$sim(D_1, D_2) = \frac{2(\sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^1 + \sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^2)}{\sum_{i=1}^{|D_1|} w_{1i} + \sum_{i=1}^{|D_2|} w_{2i}} \quad (5)$$

The value which is computed by equation (5), is always given as a normalized value between zero and one, as shown in equation (6),

$$0 \leq sim(D_1, D_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a table by the text indexing, the word weighting, and the entry selection. A table is expressed as a set of entries, and a table is filtered into a set of words by the defined function. The similarity between tables is computed based on shared words by equation (5). In the future research, we consider representing a text into multiple tables.

B. Table based KNN Algorithm

This section is concerned with the table based KNN algorithm. It was initially proposed as the approach to the text classification by Jo in 2015 [2]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \dots, T_N\}$. A novice word is encoded into a table notated by T . Its similarities with the tables which represent the sample words are computed by equation (??), as $sim(T, T_1), sim(T, T_2), \dots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

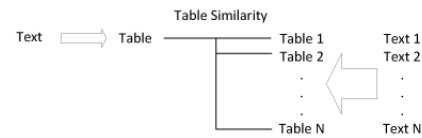


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples

are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = \text{Select}_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (8),

$$Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\} \quad (8)$$

The elements in the set, Nr , are used for voting their labels in the next step.

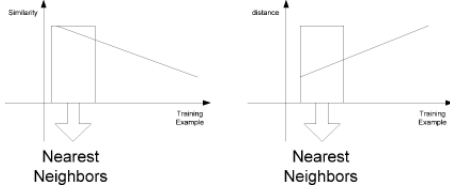


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(T_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, T , is decided by equation (9),

$$\text{label}(T) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), is called voting.

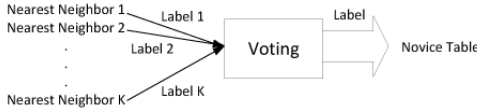


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the table based KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the text summarization system which adopts the table based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the text summarization. It is viewed as a binary classification which classifies a paragraph into summary or non-summary. This section is intended to describe the text summarization system with respect to its function and its architecture.

The process of sampling paragraphs domain by domain is illustrated in Figure 6. Even if it is possible map the text summarization into an instance of text classification, a paragraph may be classified differently depending on the domain. Sample paragraphs which are labeled with summary or non-summary are gathered domain by domain. The text which is given as the input is tagged with a domain, and the classifier which corresponds to the domain is appointed. The sample paragraphs are encoded into tables in each domain.

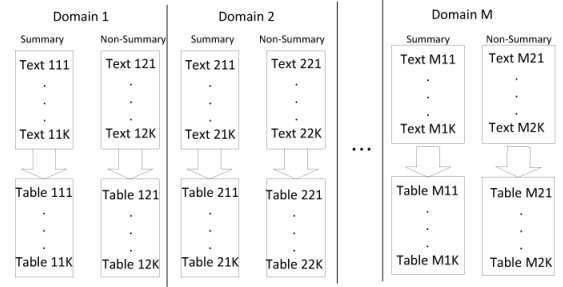


Figure 6. Preparation of Training Examples

The entire architecture of the proposed text summarization system is illustrated in Figure 7. A text is given as the input, and paragraphs are extracted by partitioning the text. In the encoding module, the sample paragraphs in the summary group and the non-summary group and ones which are extracted from the text are mapped into tables in the encoding module. The paragraphs from the text are classified into one of the two categories in the similarity computation module and the voting module. The paragraphs which are classified into summary are extracted as the output in this system.

The execution flow of the text summarization system is illustrated in Figure 8. In each domain, sample paragraphs which are labeled with summary or non-summary are gathered. The input text is partitioned into novice paragraphs, and both sample paragraphs and novice paragraphs are encoded into tables. Each paragraph is classified into summary or non-summary, and there are two groups of paragraphs in the text: summary group and non-summary group. The paragraph which are classified into summary are extracted as the summary of the input text in this system.

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. The text

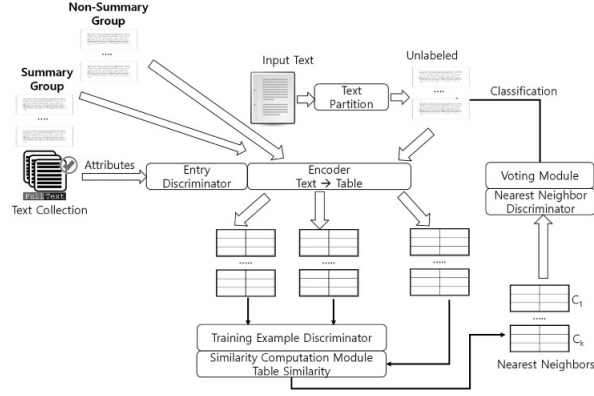


Figure 7. System Architecture

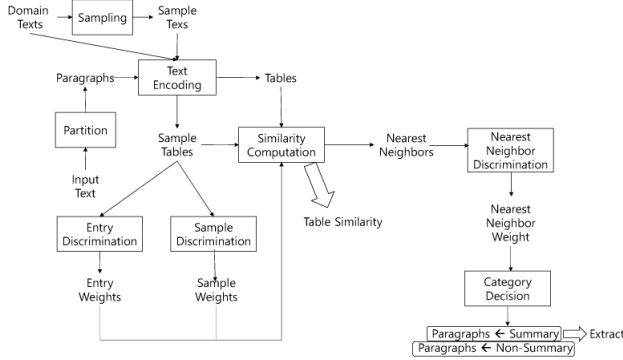


Figure 8. System Flow

summarization is defined as the binary classification where each paragraph is classified into summary or non-summary. Paragraphs are encoded into tables instead of numerical vectors, and they are classified directly. Ones which are classified with summary are selected as the summary of the input text. We need to add the text classification module for predicting the domain of input text automatically.

D. Connection with Text Classification

This section is concerned with the connection of the text summarization with the text classification. In the previous section, we mentioned the text summarization system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for implementing the text classification. This section is intended to describe the connection of the text summarization system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into

tables by the process which is described in Section II-A. The similarity computation module is for computing the similarities between tables which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

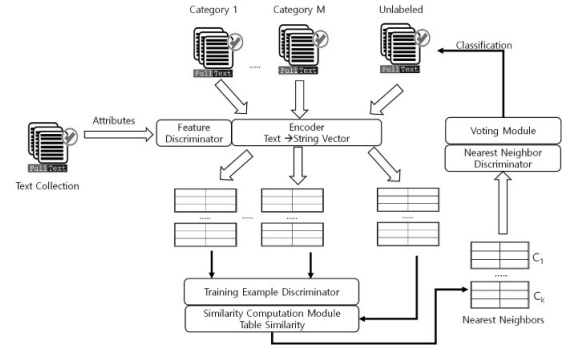


Figure 9. Text Categorization System

The connection of the text summarization with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the text summarization system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the text summarization, automatically.

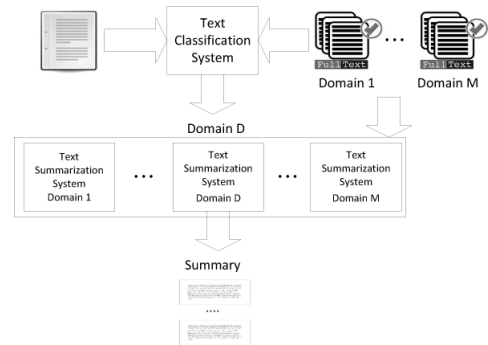


Figure 10. Text Summarization System connected with Text categorization

The process of executing the text summarization, connecting it with the text classification is illustrated in Figure 11. The sample texts each of which is labeled with its own domain are prepared, and the sample paragraphs each of which is labeled with summary or non-summary are prepared in each domain. A novice text is classified into

one among the predefined domains, and the classifier which corresponds to the domain is nominated. The novice text is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. The paragraphs which are labeled with summary is the final output from this system; the domain which is assigned to the input text is the guide for selecting a classifier.

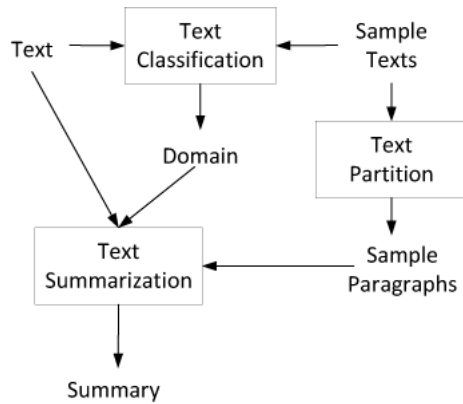


Figure 11. System Flow of Text Summarization connected with Text Categorization

Let us make some remarks on the connection of the text summarization with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the text summarization is to extract important paragraphs as the summary. In the execution flow, the input text is classified into a domain, it is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. We consider connecting the text classification as the main task the text summarization as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, 11, 2008.
- [2] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, 19, 2015.
- [3] T. Jo, "Automatic Summarization System in Current Affair Domain by Table based K Nearest Neighbor", 115-121, The Proceedings of 2nd International Conference on Advanced Engineering and ICT-Convergence, 2019.